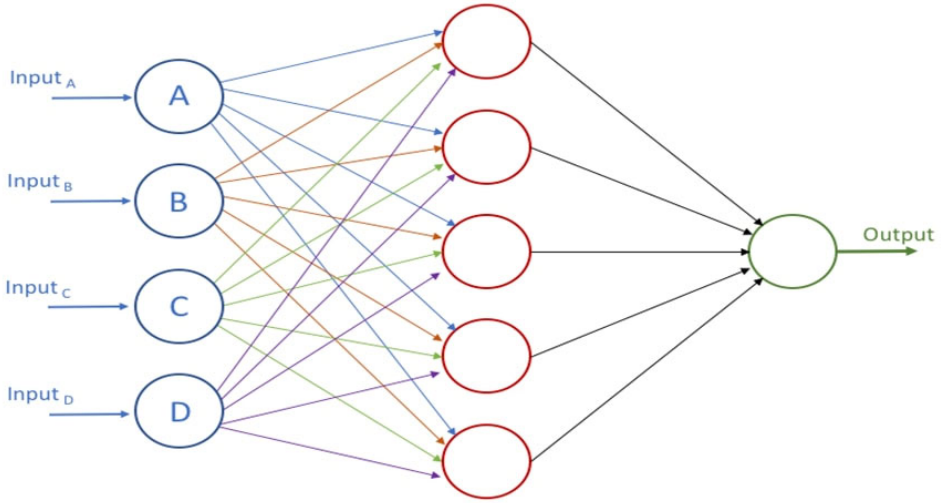
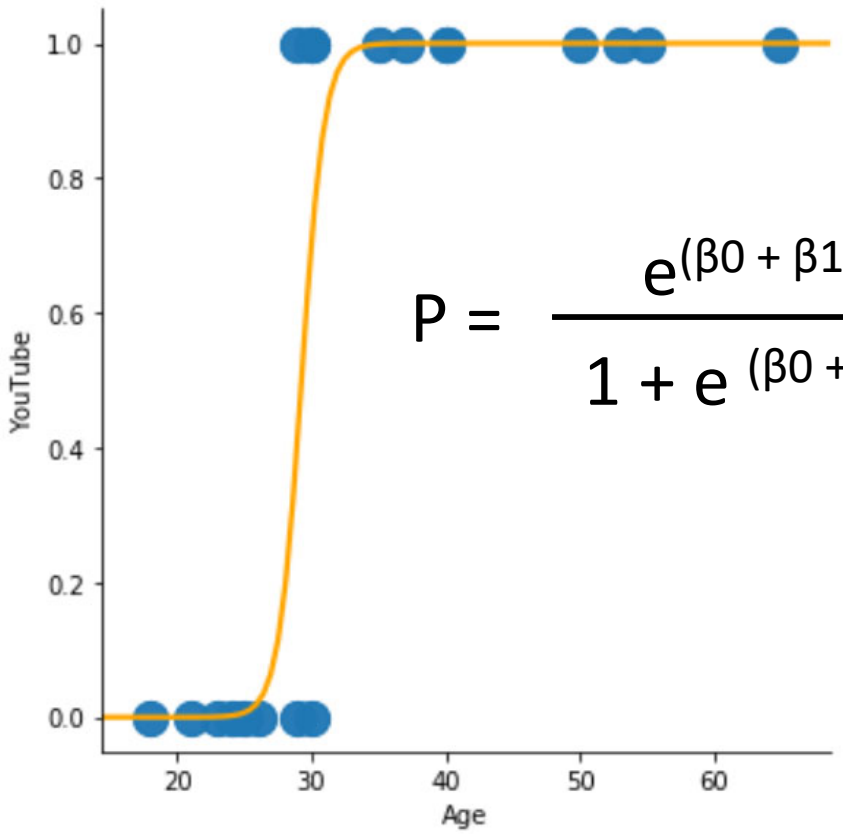
Machine Learning: Classification Problems



Neural Networks

YouTube	Age
1	30
1	35
1	40
1	50
1	65
1	55
1	53
1	40
0	30
0	23
0	21
0	18
0	25
1	30
1	29
1	37
0	24
0	26
0	29
1	30
1	35
1	40
1	50
1	65
1	55
1	53
1	40
0	30
0	23



$$P = \frac{e^{(\beta 0 + \beta 1 X1)}}{1 + e^{(\beta 0 + \beta 1 X1)}}$$

```
import mysql.connector as sq
import pandas as pd

# needed for spliting data into train and test data
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

# Connecting to MySQL, query database, store results in dataframe variable
mydb=sq.connect(host="localhost",user="root",passwd="ucla", buffered=True)
query = "SELECT * FROM youtubeage.youtubeage"
db = pd.read_sql(query,mydb)

db
```

```
# Prepare x and y
x = db[["Age"]]
y = db.YouTube
```

```
#train_and_test_data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x, y, test_size=0.2,random_state=40)
```

```
from sklearn.linear_model import LogisticRegression
logmodel = LogisticRegression(solver="lbfgs")
logmodel.fit(x_train,y_train)
```

```
y_predict=logmodel.predict(x_test)
print(y_predict)
```

```
print(y_test)
```

```
logmodel.score(x_test, y_test)
```

```
from sklearn.metrics import classification_report as report

print(report(y_test, y_predict))
```

Precision – Accuracy of positive predictions

Recall: Fraction of positives that were correctly identified

A system with high recall but low precision returns many results, but most of its predicted labels are incorrect when compared to the training labels.

A system with high precision but low recall is just the opposite, returning very few results, but most of its predicted labels are correct when compared to the training labels.

An ideal system with **high precision and high recall** will return many results, with all results labeled correctly.

F1 score – What percent of positive predictions were correct?

Support is the number of actual occurrences of the class in the specified dataset


Python Web Services

Python Data Gathering using Tiingo API

pip install tiingo

from tiingo import TiingoClient

Python Data Gathering using pandas_datareader

```
# pip install pandas-datareader

import pandas_datareader.data as rdr

import datetime as dt
```

```
# assign start and end dates for stock data to be collected
start = dt.datetime(2020,3,1)
end = dt.datetime(2021,6,2)


# use Pandas_datareader.data to grab stock data from stooq
df = rdr.DataReader("wmt", "stooq", start, end)
```

Python Extracting Tables from the Web

```
import pandas as pd

IRSTable =
pd.read_html('https://www.irs.gov/publications/p15t#en_US_2019_publink100020274')
```

Python Extracting Text from PDF

```
import PyPDF2 as pdf

myFile = open("starwars.pdf","rb")

PDFrdr = pdf.PdfFileReader(myFile)

PageText = PDFrdr.getPage(0)
```

Natural Language Processing

Natural Language Processing (NLP)

- A branch of artificial intelligence
- How to deal with text data
    - Gather, clean, decipher, analyze

Natural Language Processing (NLP)

```
import nltk
nltk.download("all")
```

Natural Language Processing (NLP)

```
import nltk

nltk.download("all")
```

```
[nltk_data] Downloading collection 'all'
[nltk_data]    |
[nltk_data]    | Downloading package abc to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping corpora\abc.zip.
[nltk_data]    | Downloading package alpino to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping corpora\alpino.zip.
[nltk_data]    | Downloading package biocreative_ppi to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping corpora\biocreative_ppi.zip.
[nltk_data]    | Downloading package brown to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping corpora\brown.zip.
[nltk_data]    | Downloading package brown_tei to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping corpora\brown_tei.zip.
[nltk_data]    | Downloading package cess_cat to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping corpora\cess_cat.zip.
```

Natural Language Processing (NLP)

```
[nltk_data]    |   Unzipping misc\perluniprops.zip.
[nltk_data]    | Downloading package nonbreaking_prefixes to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping corpora\nonbreaking_prefixes.zip.
[nltk_data]    | Downloading package vader_lexicon to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    | Downloading package porter_test to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping stemmers\porter_test.zip.
[nltk_data]    | Downloading package wmt15_eval to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping models\wmt15_eval.zip.
[nltk_data]    | Downloading package mwa_ppdb to
[nltk_data]    |     C:\Users\bhuang\AppData\Roaming\nltk_data...
[nltk_data]    |   Unzipping misc\mwa_ppdb.zip.
[nltk_data]    |
[nltk_data]  Done downloading collection all
```

- https://www.nltk.org/install.html

Natural Language Processing

Natural Language Processing

```
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
from nltk import FreqDist
```

Lion King Review on Amazon Video

The animators deserve five stars as the scenery and animals are lovely. I had to keep reminding myself that this was animated and not a live action film. Only the very first opening scene was filmed. Otherwise, the entire movie is the work of animators. Amazing. However, as a story it really falls flat. I was very disappointed. Several songs are changed, and many of the iconic scenes are scaled way back and lose their charm completely. Can You Feel The Love Tonight, during the day??? Late afternoon at best. Weird. I have four kids, they were all disappointed except the two year old who thought it was pretty great. At other points I thought it would build up to a good pun or exciting scene and it would just fall flat instead. Missed opportunities all over. The hyenas weren't funny at all. I hoped that it would pick up with Timon and Pumbaa's arrival, but it didn't. They were definitely cuter than the rest, but Timon was a real disappointment, not very funny at all. Pumbaa was actually really cute, but they got rid of, They call me Mr. Pig! So sad. I love that part. Otherwise, the animals' faces are all pretty blank for the entire movie, so the emotive acting is really unbelievable, and at times feels super overworked. The sad parts are awkward because the voice acting doesn't match the lions' relatively blank faces. Overall we all kinda hated the movie, and definitely won't watch it again. Maybe the animation should have been narrated by David Attenborough. The story would have probably been more interesting than talking realistic animals with blank stares.

Lion King Review on Amazon Video

```
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
from nltk import FreqDist
```

```
review="The animators deserve five stars as the scenery and animals are lovely. I had to keep remindin
```

```
#Tokenize words and sentences from review

word_token = word_tokenize(review.lower())

print(word_token)
```

Lion King Review on Amazon Video

```
# filter out stopwords

mystopwords = set(stopwords.words("english"))

filtered_words = []
for word in word_token:
    if word not in mystopwords and word.isalpha():
        filtered_words.append(word)

print(filtered_words)
```

```
# stemming - eliminating affixes (suffixed, prefixes, infixes, circumfixes) to obtain a word stem

ps = PorterStemmer()

stemmed_words=[]
for word in filtered_words:
    stemmed_words.append(ps.stem(word))

print("stemmed words: ",stemmed_words)
```

Lion King Review on Amazon Video

```
# Calculating frequency of words
freq = FreqDist(stemmed_words)

for word, frequency in freq.most_common(5):
      print("{}:{}".format(word, frequency))

#plot frequency for top 10
freq.plot(10,title="Top 10 from Review", linewidth=10, color="g")
```

Natural Language Processing

NLP Lemmatizer

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

Word = "skies"

print(lemmatizer.lemmatize(Word))
```